



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/975,690	10/11/2001	William W. Lee	NY-THEOR 203.1-US	2373
24972	7590	09/09/2005	EXAMINER	
FULBRIGHT & JAWORSKI, LLP 666 FIFTH AVE NEW YORK, NY 10103-3198			CHOUDHURY, AZIZUL Q	
			ART UNIT	PAPER NUMBER
			2145	

DATE MAILED: 09/09/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

**Office Action Summary**

Application No.

09/975,690

Applicant(s)

LEE ET AL.

Examiner

Azizul Choudhury

Art Unit

2145

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --  
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☒ Responsive to communication(s) filed on 07 June 2005.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 11 October 2001 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
  - ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date 2/02.
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_.

Art Unit: 2145

***Detailed Action***

This office action is in response to the correspondence received on June 7, 2005.

***Response to Amendment***

Applicant's request for reconsideration of the finality of the rejection of the last Office action is persuasive and, therefore, the finality of that action is withdrawn.

***Claim Rejections - 35 USC § 112***

The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

Claim 20 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. The phrase "round trip engineering support," is vague and unclear.

***Claim Rejections - 35 USC § 103***

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

Claims 1-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Iyengar et al (US Pat No: US006018627A) in view of Anne Thomas ("Container-Managed Persistence"), hereafter referred to as Iyengar and Thomas, respectively.

1. With regards to claim 1, Iyengar teaches through Thomas, a method of generating code for Enterprise JavaBean (EJB) components from a business process, comprising the steps of graphically modeling said business process using a UML drawing tool to provide an UML model having a plurality of EJB Classes; defining relationships between said plurality of EJB classes; stereotyping each of said plurality of EJB classes into one or more EJB components; transforming each of said EJB components into EJB source code; and embedding code marker; in said EJB source code to enable subsequent updates to said EJB source code

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Iyengar's design also allows for business logic (Figure 3, Iyengar), which is equivalent to the claimed embedding code marker. However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within

persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development.

Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

2. With regards to claim 2, Iyengar teaches through Thomas, the method further comprising the step of compiling said EJB source code to generate EJB application in accordance with deployment properties

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Iyengar also discloses how source code is compiled (build) to generate applications in accordance with deployment properties (Figure 3, item 34, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within

persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

3. With regards to claim 3, Iyengar teaches through Thomas, a method further comprising the step of deploying said EJB application to a server using one of the following: bean managed persistence or container managed persistence

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that persistence or container managed persistence is available (p. 1, Thomas). Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a

database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

4. With regards to claim 4, Iyengar teaches through Thomas, a method wherein the step of stereotyping stereotypes an EJB class into at least one of the following Smart EJB component: Belonging, Session, Entity, Configurable Entity, Business Policy and Workflow

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is

applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent. In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

5. With regards to claim 5, Iyengar teaches through Thomas, a method wherein an Entity EJB component comprises at least one interface and two EJB classes

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to



ensure when an instance of an object is run, the code it is mapped to is recent. In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

6. With regards to claim 6, Iyengar teaches through Thomas, the method wherein said Entity EJB component being associated with a Primary Key class and a Value class

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to

Art Unit: 2145

ensure when an instance of an object is run, the code it is mapped to is recent.

In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available, especially since Java is an object oriented language.

Both Iyengar and Thomas teach designs for software development.

Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

7. With regards to claim 7, Iyengar teaches through Thomas, the method wherein each EJB component includes at least one of the following: name, stereotype, attribute and method

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to

Art Unit: 2145

ensure when an instance of an object is run, the code it is mapped to is recent.

In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available.

Both Iyengar and Thomas teach designs for software development.

Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

8. With regards to claim 8, Iyengar teaches through Thomas, the method wherein each attribute includes a pair of accessor methods

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available, in particular, it corresponds to EJB object interface used by the client to access the business method within the object (p. 3, Thomas).

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

9. With regards to claim 9, Iyengar teaches through Thomas, the method wherein said relationships includes at least one of the following: inheritance and aggregation

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to

ensure when an instance of an object is run, the code it is mapped to is recent.

In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available, in particular it corresponds to extending a preexisting object class for new functionality (inheritance) and simple containment of another object (aggregation) (p. 7, Thomas).

Both Iyengar and Thomas teach designs for software development.

Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

10. With regards to claim 10, Iyengar teaches through Thomas, the method wherein said aggregation includes multiplicity

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is

applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent. In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available, in particular it corresponds to when an object could point to hundreds of other objects (p. 7, Thomas).

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

11. With regards to claim 11, Iyengar teaches through Thomas, a method further comprising the steps of: determining if said multiplicity relationship is one to many; and stereotyping said aggregation relationship into a collection type if it is determined that said multiplicity relationship is one to many

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, Iyengar discloses how relationships such as aggregation is permitted (column 4, lines 3-11, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent. In addition, the claimed trait is a trait of EJB and Thomas teaches how EJB is available, in particular it corresponds to object relationships (p. 7, Thomas).

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

12. With regards to claim 12, Iyengar teaches through Thomas, the method wherein said collection type includes one of the following: Set, Array, List or Map

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, Iyengar discloses the use of repositories (collection type) (column 4, lines 21-26, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

13. With regards to claim 13, Iyengar teaches through Thomas, the method wherein each EJB component is a Smart Component having at least one Smart Feature (Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, since any language is permissible (including Java, an object oriented language), classes are acceptable for the design and hence means are present by which to provide the claimed traits. However, Iyengar's design does not teach EJB specific traits.



Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

14. With regards to claim 14, Iyengar teaches through Thomas, the method wherein said Smart Feature includes one of the following: SmartKey, SmartHandle and SmartValue

(Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, since any language is permissible (including Java, an object oriented language), classes are acceptable for the design and hence means are present by which to provide the claimed traits. However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

15. With regards to claim 15, Iyengar teaches through Thomas, the method wherein said Smart component is an eBusiness Smart Component

(Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, since any language is permissible (including Java, an object oriented language), classes are acceptable for the design and hence means are present by which to provide the claimed traits. Furthermore, Iyengar's design is intended to allow for eBusinesses (column 3,

lines 56-65, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

16. With regards to claim 16, Iyengar teaches through Thomas, the method wherein the step of transforming includes the step generating said EJB codes according to a Code Template Dictionary

(Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, Iyengar's design allows for a

Art Unit: 2145

repository (column 4, line 26 – column 5, line 10, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

17. With regards to claim 17, Iyengar teaches through Thomas, the method wherein said Code Template Dictionary includes key-value pair entries

(Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, Iyengar's design allows for a repository (column 4, line 26 – column 5, line 10, Iyengar). The repository allows

for a variety of data to be stored. However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

18. With regards to claim 18, Iyengar teaches through Thomas, the method wherein values of said Code Template Dictionary represent EJB code templates

(Iyengar teaches a UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Plus, Iyengar's design allows for a repository (column 4, line 26 – column 5, line 10, Iyengar). The repository allows

for a variety of data to be stored. However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

19. With regards to claim 19, Iyengar teaches through Thomas, the method wherein the step of embedding includes the step of adding business logic code between said code markers

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar discloses how the design allows for any language to be incorporated with the design (column 9, lines 32-35, Iyengar). Iyengar's design also allows for

business logic (Figure 3, Iyengar), which is equivalent to the claimed embedding code marker. However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

20. With regards to claim 20, Iyengar teaches through Thomas, the method further comprising the step of synchronizing said UML model with said business logic code, thereby providing round trip engineering support

(Iyengar teaches an UML design (column 3, line 45 – column 4, line 33, Iyengar). A UML enables a user to graphically model business models along with their relationships and translate them into source code. In addition, Iyengar's design provides complete service from design composition to product deployment (Figures 2A, 2B, 10A, 10B, 10C; and 14, Iyengar). However, Iyengar's design does not teach EJB specific traits.

Thomas discloses the traits of EJB. Within the disclosure, Thomas teaches that mapping tools are available in EJB to enable persistence. Within persistence, code can be updated since the code itself is in a database and is applied only when needed. This allows for updates to the code to occur to ensure when an instance of an object is run, the code it is mapped to is recent.

Both Iyengar and Thomas teach designs for software development. Therefore, it would have been obvious to one skilled in the art, during the time of invention, to have combined the teachings of Iyengar with those of Thomas, to enable portability across application servers, component reusability and increased developer productivity (p. 1, Thomas)).

### ***Response to Remarks***

After a review of the application, finality for the case has been withdrawn and a new office action is being issued. While the Thomas art was used in the past, the Iyengar art is a newly applied art that focuses on UML. It allows for a UML design that permits many languages, including Java. Hence, it has support for classes (to support the claimed "smart" features of the claimed invention). In addition, the Iyengar prior art supports business logic and aggregation amongst significant other features.

### ***Conclusion***




Art Unit: 2145

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Azizul Choudhury whose telephone number is (571) 272-3909. The examiner can normally be reached on M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Rupal Dharia can be reached on (571) 272-3880. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AC

  
RUPAL DHARIA  
SUPERVISORY PATENT EXAMINER